

5

BACKGROUND OF THE INVENTION

The present invention relates generally to the field
10 of computer software and, more specifically, to a method,
system, and apparatus for managing addressing to
input/output drawers.

15 A logical partitioning option (LPAR) within a data
processing system (platform) allows multiple copies of a
single operating system (OS) or multiple heterogeneous
operating systems to be simultaneously run on a single
data processing system platform. A partition, within
20 which an operating system image runs, is assigned a
non-overlapping sub-set of the platform's resources.
These platform allocable resources include one or more
architecturally distinct processors with their interrupt
management area, regions of system memory, and
25 input/output (I/O) adapter bus slots. The partition's
resources are represented by its own open firmware device
tree to the OS image.

Each distinct OS or image of an OS running within the platform is protected from each other such that software errors on one logical partition can not affect the correct operation of any of the other partitions. This is provided by allocating a disjoint set of platform

REF ID: A68093

resources to be directly managed by each OS image and by providing mechanisms for ensuring that the various images can not control any resources that have not been allocated to it. Furthermore, software errors in the control of an OS's allocated resources are prevented from affecting the resources of any other image. Thus, each image of the OS (or each different OS) directly controls a distinct set of allocable resources within the platform.

10 In many systems, I/O devices are incorporated into
the data processing system using I/O drawers. These I/O
drawers are modular structures that are easy to install
and remove allowing for easy modification of the data
processing system. The I/O drawers typically contain
15 several I/O expansion slots in which I/O devices may be
"plugged" into and used by the data processing system.
For example, many I/O drawers allow for 8 or 16 I/O
expansion slots.

Each I/O drawer and expansion slot within the I/O drawer must be assigned addresses by the data processing system such that input and output requests from various components within the system may utilize the new hardware. In prior art RIO systems, open firmware peripheral component interconnect (PCI) Host Bridge (PHB) unit addresses are dynamically generated based on dynamic discovery of PHBs on successive Remote Input/Output (RIO) loop probes. A RIO system employs a special I/O bridge, which is called an RIO hub and has several ports to connect to I/O drawers via special high-speed cables. An I/O drawer has two ports. There are two typical RIO loops: 1) one port of a hub connected to the input port

Docket No. AUS9-2000-0316-US1

00631723-080300

of an I/O drawer, and the output port of this I/O drawer connected to the companion port of the same hub; 2) one port of a hub connected to the input port of an I/O drawer, the output port of this I/O drawer connected to the input port of another I/O drawer, and the output port of the other I/O drawer connected to the companion port of the same hub. An RIO loop probe refers to the discovery process to determine the presence of one of these two RIO loops. The ODM of some operating systems, such as, for example, Advanced Interactive Executive (AIX) operating system, use the open firmware device path (e.g. /pci@fba0000000/scsi) as the identifier of an Object Database Management (ODM) object. ODM is a software component of AIX. Hardware functional components such as PCI Host Bridges (PHBs) are represented as ODM objects in the database to be managed by the ODM software. If a user moves an RIO drawer from one RIO loop to another, all open firmware PHB unit addresses change. The AIX ODM then presents the user with questions regarding the disappearance of the associated resources for the "old" drawer and the appearance of resources associated with a "new" drawer, when, from the user's point of view, all of the same resources are still being employed. Currently, the user must manually resolve the AIX ODM when the drawer is moved to a different location within the same data processing system.

Such occurrences can be confusing and annoying to users, therefore, a data processing system in which I/O drawers may be inserted, removed, and rearranged without requiring a user to resolve any address problems would be

desirable.

0967-836X(199609)14:3<231::AID-JCOP231>3.0.CO;2-D

Docket No. AUS9-2000-0316-US1

SUMMARY OF THE INVENTION

5 The present invention provides a method, system, and apparatus for managing input/output drawers within a data processing system. In one embodiment, a unique location identifier is assigned to each of a plurality of input/output drawers. The unique location identifier is stored in memory and is used by the operating system to identify the plurality of input/output drawers regardless of how the input/output drawers are interconnected by cables. When a new input/output drawer is added to the data processing system, the service processor assigns a new unique location identifier to the new input/output drawer ensuring that the unique location identifiers previously assigned to other I/O drawers are not used to identify the new I/O drawer.

Docket No. AUS9-2000-0316-US1

BRIEF DESCRIPTION OF THE DRAWINGS

5

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

Figure 1 depicts a block diagram of a data processing system in which the present invention may be implemented;

Figure 2 depicts a block diagram of an exemplary logically partitioned platform in which the present invention may be implemented;

Figure 3 depicts a block diagram of a system for installing and managing a system I/O drawers in accordance with the present invention;

Figure 4 depicts a flowchart illustrating an exemplary process for incorporating a new I/O drawer into a data processing system in accordance with the present invention; and

Figure 5 depicts a flowchart illustrating an exemplary process for assigning memory mapping to I/O drawers in accordance with the present invention.

00000000 "E2ZTE950

Docket No. AUS9-2000-0316-US1

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

With reference now to the figures, and in particular with reference to **Figure 1**, a block diagram of a data processing system in which the present invention may be implemented is depicted. Data processing system **100** may be a symmetric multiprocessor (SMP) system including a plurality of processors **101**, **102**, **103**, and **104** connected to system bus **106**. For example, data processing system **100** may be an IBM RS/6000, a product of International Business Machines Corporation in Armonk, New York, implemented as a server within a network. Alternatively, a single processor system may be employed. Also connected to system bus **106** is memory controller/cache **108**, which provides an interface to a plurality of local memories **160-163**. I/O bus bridge **110** is connected to system bus **106** and provides an interface to I/O bus **112**. Memory controller/cache **108** and I/O bus bridge **110** may be integrated as depicted.

Data processing system **100** is a logically partitioned data processing system. Thus, data processing system **100** may have multiple heterogeneous operating systems (or multiple instances of a single operating system) running simultaneously. Each of these multiple operating systems may have any number of software programs executing within in it. Data processing system **100** is logically partitioned such that different I/O adapters **120-121, 128-129, 136, and 148-149**

may be assigned to different logical partitions.

5

15

20

30 ~~implementations will support between four and eight I/O~~

Docket No. AUS9-2000-0316-US1

Pro A3
~~adapters (i.e. expansion slots for add-in connectors).~~

Each I/O Adapter **120-121** provides an interface between data processing system **100** and input/output devices such as, for example, other network computers, which are

5 ~~clients to data processing system 100.~~ *Pro A3*

Pro A4
~~An additional PCI host bridge **122** provide an interface for an additional PCI bus **123**. PCI bus **123** is connected to a plurality of PCI I/O adapters **128-129** by a PCI bus **126-127**. Thus, additional I/O devices, such as, for example, modems or network adapters may be supported through each of PCI I/O adapters **128-129**. In this manner, data processing system **100** allows connections to multiple network computers.~~ *Pro A4*

10
 15 A memory mapped graphics adapter **148** may be connected to I/O bus **112** through PCI Host Bridge **140** and EADS **142** (PCI-PCI bridge) via PCI buses **141** and **144** as depicted. Also, a hard disk **150** may also be connected to I/O bus **112** through PCI Host Bridge **140** and EADS **142** via PCI buses **141** and **145** as depicted.

Pro A5 20
~~A PCI host bridge **130** provides an interface for a PCI bus **131** to connect to I/O bus **112**. PCI bus **131** connects PCI host bridge **130** to the service processor mailbox interface and ISA bus access pass-through logic **194** and EADS **132**. The ISA bus access pass-through logic **194** forwards PCI accesses destined to the PCI/ISA bridge **193**. The NV-RAM storage is connected to the ISA bus **196**. The Service processor **135** is coupled to the service processor mailbox interface **194** through its local PCI bus **195**. Service processor **135** is also connected to processors **101-104** via a plurality of JTAG/I²C buses **134**.~~

25
 30

00000000 "C" 247E960

Docket No. AUS9-2000-0316-US1

hops

JTAG/I²C buses **134** are a combination of JTAG/scan busses (see IEEE 1149.1) and Phillips I²C busses. However, alternatively, JTAG/I²C buses **134** may be replaced by only Phillips I²C busses or only JTAG/scan busses. All

5 SP-ATTN signals of the host processors **101**, **102**, **103**, and **104** are connected together to an interrupt input signal of the service processor. The service processor **135** has its own local memory **191**, and has access to the hardware

~~bp-panel 190-*15*~~

10 When data processing system **100** is initially powered up, service processor **135** uses the JTAG/scan buses **134** to interrogate the system (Host) processors **101-104**, memory controller **108**, and I/O bridge **110**. At completion of this step, service processor **135** has an inventory and

15 topology understanding of data processing system **100**. Service processor **135** also executes Built-In-Self-Tests (BISTs), Basic Assurance Tests (BATs), and memory tests on all elements found by interrogating the system

processors **101-104**, memory controller **108**, and I/O bridge

20 **110**. Any error information for failures detected during the BISTs, BATs, and memory tests are gathered and reported by service processor **135**.

If a meaningful/valid configuration of system resources is still possible after taking out the elements

25 found to be faulty during the BISTs, BATs, and memory tests, then data processing system **100** is allowed to proceed to load executable code into local (Host) memories **160-163**. Service processor **135** then releases the Host processors **101-104** for execution of the code

30 loaded into Host memory **160-163**. While the Host

00000000-00000000

processors **101-104** are executing code from respective operating systems within the data processing system **100**, service processor **135** enters a mode of monitoring and reporting errors. The type of items monitored by service processor include, for example, the cooling fan speed and operation, thermal sensors, power supply regulators, and recoverable and non-recoverable errors reported by processors **101-104**, memories **160-163**, and bus-bridge controller **110**.

Service processor **135** is responsible for saving and reporting error information related to all the monitored items in data processing system **100**. Service processor **135** also takes action based on the type of errors and defined thresholds. For example, service processor **135** may take note of excessive recoverable errors on a processor's cache memory and decide that this is predictive of a hard failure. Based on this determination, service processor **135** may mark that resource for deconfiguration during the current running session and future Initial Program Loads (IPLs). IPLs are also sometimes referred to as a "boot" or "bootstrap".

Those of ordinary skill in the art will appreciate that the hardware depicted in **Figure 1** may vary. For example, other peripheral devices, such as optical disk drives and the like, also may be used in addition to or in place of the hardware depicted. The depicted example is not meant to imply architectural limitations with respect to the present invention.

~~With reference now to **Figure 2**, a block diagram of~~
~~an exemplary logically partitioned platform is depicted~~

John A

5

15

25

30

Docket No. AUS9-2000-0316-US1

images **202-208** by virtualizing all the hardware resources of logically partitioned platform **200**. OF **210** may attach I/O devices through I/O adapters **248-262** to single virtual machines in an exclusive mode for use by one of
5 OS images **202-208**.

With reference now to **Figure 3**, a block diagram of a system for installing and managing a system I/O drawers is depicted in accordance with the present invention. System **300** may be implemented within a data processing
10 system such as, for example, logically partitioned platform **200** in **Figure 2**. A system I/O drawer is a modular component for inserting I/O expansion slots into a data processing system. An I/O drawer physically packages several PHBs to provide PCI I/O slots for
15 plug-in I/O adapters. In **Figure 1**, everything attached to I/O bus **112** could reside in an I/O drawer, including the service processor **135**. The I/O bus **112** is a special high-speed cable connecting the I/O bridge **110**, which is called a hub, to the I/O drawer's input/output ports. The
20 I/O drawer containing the service processor **135** is called the primary drawer. All other I/O drawers are connected via the System Power Control Network (SPCN) bus **380** to the service processor **135**.

System **300** includes three I/O drawers **304-308**. Each
25 of I/O drawers **304-308** contains two PCI host bridges (PHBs) **310-320**. However, although depicted with three I/O drawers **304-308** and two PHBs **310-320**, one skilled in the art will recognize that more or fewer I/O drawers and PHBs may be included than depicted in **Figure 3**. Each PHB
30 **310-320** may support, for example, between 8 and 16 PCI

005080"E24TE960

expansion slots, which may be implemented, for example, as I/O adapters **248-262** in **Figure 2**.

Service processor **302**, which may be implemented, for example, as service processor **290** in **Figure 2**, assigns a unique SPCN ID to each of I/O drawers **304-308** within the system **300**. Service processor uses the SPCN bus to detect and assign unique IDs to I/O drawers, to control the power logic of the I/O drawers, and to monitor their environmental sensors such as drawer temperature, fan speed, etc. The SPCN ID is then associated with the drawer's unique serial number from the drawer's Vital Product Data (VPD). The VPD contains information related to the product in which it is found such as, for example, product manufacturer, product serial number, and part number. When a new drawer is added to system **300**, service processor **302** changes the SPCN ID of the new drawer to a value not being used by any of the existing I/O drawers **304-308**. An SPCN/SN table **324** within NVRAM **322** is updated by service processor **302** to reflect the new assignment of the SPCN ID. NVRAM **322** may be implemented as, for example, NVRAM **298** in **Figure 2**. The SPCN/SN table **324** is used in determining if a new I/O drawer is installed since the new I/O drawer's serial number is not in the existing table. From the SPCN/SN table **324**, the service processor **302** can find out all SPCN IDs currently used by the existing I/O drawers **304-308** so that it can select an unused SPCN ID for the new drawer. The SPCN ID can be used to label an I/O drawer by displaying its SPCN ID to the I/O drawer LCD operator panel.

System firmware **326**, which may be implemented as open firmware **210** in **Figure 2**, dynamically discovers the I/O drawers **304-308** and assigned memory mapping to each one of drawers **304-308** and its PHBs **310-320**. The

20 Firmware **326** also creates location codes for PHB
nodes as, for example, U0.X-P1 where X is the SPCN ID of
the drawer. The device nodes and location codes are
stored in open firmware (OF) device tree **342** within
system memory **340**. System memory **340** may be implemented
25 as, for example, memory **191** in **Figure 1**. The PHB nodes
are parts of the open firmware device tree **342**
constructed by open firmware in system memory **340**. Since
the serial number and SPCN ID are permanently associated
and maintained by service processor **302**, the ODM of the
30 OS, such as, for example, one of OSs **202-208**, will be

unchanged for the drawer. AIX Object Database Management (ODM) software allows users to query and manage system configuration.

Those of ordinary skill in the art will appreciate that the components depicted in **Figure 3** may vary. For example, other I/O adapters may be utilized rather than PCI type adapters. The depicted example is not meant to imply architectural limitations with respect to the present invention.

With reference now to **Figure 4**, a flowchart

Docket No. AUS9-2000-0316-US1

illustrating an exemplary process for incorporating a new I/O drawer into a data processing system is depicted in accordance with the present invention. After a new I/O drawer, such as, for example, one of I/O drawers **304-308** is inserted into a data processing system, the service processor recognizes that a new drawer has been installed (step **402**). The service processor then consults an SPCN/SN table of assignments of SPCN IDs to existing I/O drawers to ensure that the new I/O drawer is not assigned an SPCN previously assigned to another I/O drawer (step **404**). The service processor then changes the new I/O drawer's SPCN ID to one not being used by one of the existing I/O drawers (step **406**). The SPCN/SN table is then updated to reflect the new assignment (step **408**).

With reference now to **Figure 5**, a flowchart illustrating an exemplary process for assigning memory mapping to I/O drawers is depicted in accordance with the present invention. As I/O drawers, such as, for example, one of I/O drawers **304-308** in **Figure 3**, are added to a data processing system, the system firmware, such as, for example, firmware **326** in **Figure 3**, discovers the I/O drawers (step **502**). The firmware then assigns memory mapping to the I/O drawer and each of its associated PHBs (step **504**). Memory mapping to an I/O drawer as used herein means assigning system memory address ranges so that these addresses can be used by the host processors to access I/O devices within the drawer. SPCN ID is not involved in the memory mapping process. SPCN ID is used by open firmware to generate PHB location codes for the PHB nodes in the device tree. The firmware then creates PHB nodes (step **506**) and location codes for the PHB nodes

(step **508**) and stores this information in the open firmware device tree (step **510**) located in system memory, such as, for example, system memory **340** in **Figure 3**.

5 PHB nodes are device representation of the PHB hardware
in the open firmware device tree. The nodes contain the
open firmware device properties which describe the
characteristics of the PHBs and the memory mapping of the
PHBs, and open firmware methods which are software device
10 driver function for the PHBs.

It is important to note that while the present invention has been described in the context of a fully functioning data processing system, those of ordinary skill in the art will appreciate that the processes of the present invention are capable of being distributed in the form of a computer readable medium of instructions and a variety of forms and that the present invention applies equally regardless of the particular type of signal bearing media actually used to carry out the distribution. Examples of computer readable media include recordable-type media such a floppy disc, a hard disk drive, a RAM, and CD-ROMs and transmission-type media such as digital and analog communications links.

The description of the present invention has been presented for purposes of illustration and description, but is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. The embodiment was chosen and described in order to best explain the principles of the invention, the practical application, and to enable others of

ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated.